

**Kauno technologijos universitetas**

Informatikos fakultetas

**Objektinis programavimas I (P175B118)**

Laboratorinių darbų ataskaita

**Vytenis Kriščiūnas IFF-1/1**

**Docentas Giedrius Ziberkas**

**Kaunas 2021**

TURINYS

[**1. Duomenų klasė 3**](#_Toc83069925)

[1.1. Darbo užduotis 3](#_Toc83069926)

[1.2. Programos tekstas 3](#_Toc83069927)

[1.3. Pradiniai duomenys ir rezultatai 8](#_Toc83069928)

[1.4. Dėstytojo pastabos 9](#_Toc83069929)

# Duomenų klasė

## Darbo užduotis

Krepšinio rinktinė. Artėja Pasaulio vyrų krepšinio čempionatas. Turime į rinktinės stovyklą pakviestų kandidatų sąrašą. Duomenų faile pateikiama informacija apie pakviestus krepšininkus: vardas, pavardė, gimimo data, ūgis, pozicija, klubas, požymis „pakviestas“, požymis „kapitonas“ (true, false).

• Raskite jauniausią į rinktinę pakviestą krepšininką, ekrane atspausdinkite jo vardą, pavardę, amžių ir poziciją. Jei yra keli, spausdinkite visus.

• Raskite krepšininkus, žaidusius Kauno „Žalgiryje“, ekrane atspausdinkite jų vardus, pavardes bei pozicijas.

• Krepšininkai mėgsta švęsti gimtadienius. Sudarykite sąrašą krepšininkų, kurie švęs gimtadienius pasirengimo krepšinio čempionatui metu (liepos 20d. – rugsėjo 3d.), į failą „Gimtadieniai.csv“ įrašykite krepšininkų vardus, pavardes bei gimimo mėnesį ir dieną.

## Programos tekstas

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace \_13\_uzduotis

{

//Class that calculates given information and forms lists

class TaskUtils

{

/// <summary>

/// Creates a list to disperse the information

/// </summary>

/// <param name="players">Array of players</param>

/// <returns>Formated list</returns>

public static List<Player> Youngest(List<Player> players)

{

List<Player> youngest = new List<Player>();

Player age = players[0];

for (int i = 1; i < players.Count; i++)

{

if (DateTime.Compare(players[i].BirthDate, age.BirthDate) > 0) //Searching for earliest DateTime information

{

age = players[i];

}

}

for (int i = 0; i < players.Count; i++)

{

if (age.BirthDate == players[i].BirthDate) //Comparing ealiest DateTime information to player birth date

{

youngest.Add(players[i]);

}

}

return youngest;

}

/// <summary>

/// Creates a list to disperse the information

/// </summary>

/// <param name="players">Array of players</param>

/// <param name="team">string representing a team</param>

/// <returns>Formated list</returns>

public static List<Player> Zalgiris(List<Player> players, string team)

{

List<Player> InTheTeam = new List<Player>();

foreach (Player player in players)

{

if (player.Club.Equals(team))

{

InTheTeam.Add(player);

}

}

return InTheTeam;

}

/// <summary>

/// Creates a list to disperse the information

/// </summary>

/// <param name="players">Array of players</param>

/// <returns>Formated list</returns>

public static List<Player> CelebratesBirthDays(List<Player> players)

{

List<Player> Celebrates = new List<Player>();

DateTime DateBegining = new DateTime(DateTime.Now.Year, 7, 20); //Intodusing new DateTime variable

DateTime DateEnding = new DateTime(DateTime.Now.Year, 9, 3); //Intodusing new DateTime variable

foreach (Player player in players)

{

if (player.BirthDate.DayOfYear >= DateBegining.DayOfYear && player.BirthDate.DayOfYear <= DateEnding.DayOfYear) //Converting DateTime information to values and then comparing them

{

Celebrates.Add(player);

}

}

return Celebrates;

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace \_13\_uzduotis

{

//Class that saves information about one basketball player

class Player

{

public string Name { get; set; }

public string Surname { get; set; }

public DateTime BirthDate { get; set; }

public int Hight { get; set; }

public int Number { get; set; }

public string Club { get; set; }

public bool Invited { get; set; }

public bool CaptainOrNot { get; set; }

/// <summary>

/// Creates public method with the same name as class name

/// </summary>

/// <param name="name">Name of player</param>

/// <param name="surname">Surname of player</param>

/// <param name="birthDate">Birth date of player</param>

/// <param name="hight">Hight of player</param>

/// <param name="number">Number of player</param>

/// <param name="club">Club of player</param>

/// <param name="invited">Invited or not invited player</param>

/// <param name="captainOrNot">Player who is captain or not</param>

public Player(string name, string surname, DateTime birthDate, int hight, int number, string club, bool invited, bool captainOrNot)

{

this.Name = name;

this.Surname = surname;

this.BirthDate = birthDate;

this.Hight = hight;

this.Number = number;

this.Club = club;

this.Invited = invited;

this.CaptainOrNot = captainOrNot;

}

/// <summary>

/// Creates int method

/// </summary>

/// <returns>Formated int value</returns>

public int CalculateAge()

{

DateTime today = DateTime.Today;

int age = today.Year - this.BirthDate.Year;

if (this.BirthDate.Date > today.AddYears(-age))

{

age--;

}

return age;

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.IO;

using System.Threading.Tasks;

namespace \_13\_uzduotis

{

//Class that prints or reads information

class InOutUtils

{

/// <summary>

/// Creates a list to disperse the information

/// </summary>

/// <param name="fileName">Specific file name</param>

/// <returns>Formated list</returns>

public static List<Player> ReadFile(string fileName)

{

List<Player> Players = new List<Player>();

string[] Lines = File.ReadAllLines(fileName, Encoding.UTF8);

foreach (string line in Lines)

{

string[] Values = line.Split(';');

string name = Values[0];

string surname = Values[1];

DateTime birthDate = DateTime.Parse(Values[2]);

int hight = int.Parse(Values[3]);

int number = int.Parse(Values[4]);

string club = Values[5];

//Finding out if player is invited or not

bool Invited = false;

if (Values[6] == "pakviestas")

{

Invited = true;

}

//Finding out if player is captain or not

bool captainOrNot = false;

if (Values[7] == "kapitonas")

{

captainOrNot = true;

}

Player Player = new Player(name, surname, birthDate, hight, number, club, Invited, captainOrNot);

Players.Add(Player);

}

return Players;

}

/// <summary>

/// Creates a void function where information is printed

/// </summary>

/// <param name="players">Array of players</param>

/// <param name="fileName1">Specific file name</param>

public static void PrintToTxt(List<Player> players, string fileName1)

{

string[] lines = new string[players.Count + 4];

lines[0] = String.Format(new string('-', 121));

lines[1] = String.Format("| {0, -8} | {1, -12} | {2, -6} | {3, 8} | {4, 8} | {5, -8} | {6, -8} | {7, -8} |", "Vardas", "Pavardė", "Gimimo data", "Žaidėjo ūgis", "Numeris", "Klubas", "Ar pakviestas", "Komandos kapitonas ar ne");

lines[2] = String.Format(new string('-', 121));

for (int i = 0; i < players.Count; i++)

{

lines[i + 3] = String.Format("| {0, -8} | {1, -12} | {2, -11:yyyy-MM-dd} | {3, 12} | {4, 8} | {5, -8} | {6, -13} | {7, -24} |", players[i].Name, players[i].Surname, players[i].BirthDate, players[i].Hight, players[i].Number, players[i].Club, players[i].Invited, players[i].CaptainOrNot);

}

lines[players.Count + 3] = String.Format(new string('-', 121));

File.WriteAllLines(fileName1, lines, Encoding.UTF8);

}

/// <summary>

/// Creates a void function where information is printed

/// </summary>

/// <param name="players">Array of players</param>

public static void PrintYoungestPlayers(List<Player> players)

{

foreach (Player player in players)

{

Console.WriteLine("{0};{1};{2};{3}", player.Name, player.Surname, player.CalculateAge(), player.Number);

}

}

/// <summary>

/// Creates a void function where information is printed

/// </summary>

/// <param name="players">Array of players</param>

public static void PrintClubPlayers(List<Player> players)

{

foreach (Player player in players)

{

Console.WriteLine("{0};{1};{2}", player.Name, player.Surname, player.Number);

}

}

/// <summary>

/// Creates a void function where information is printed

/// </summary>

/// <param name="fileName">Specific file name</param>

/// <param name="players">Array of players</param>

public static void PrintToCsv(string fileName, List<Player> players)

{

string[] lines = new string[players.Count];

for (int i = 0; i < players.Count; i++)

{

lines[i] = string.Format("{0};{1};{2:MM-dd}", players[i].Name, players[i].Surname, players[i].BirthDate);

}

File.WriteAllLines(fileName, lines, Encoding.UTF8);

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.IO;

using System.Threading.Tasks;

//Main function of this program is to do all kinds of calculations with different basketball players information

//Vytenis Kriščiūnas

namespace \_13\_uzduotis

{

//Main class

class Program

{

const string CFd = @"Players.txt"; //Represents a .txt file from which data will be read

const string CFr1 = "Rezults.txt"; //Represents a .txt file where data will be put

const string CFr2 = "Gimtadieniai.csv"; //Represents a .csv file where data will be put

static void Main(string[] args)

{

List<Player> allPlayers = InOutUtils.ReadFile(CFd);

InOutUtils.PrintToTxt(allPlayers, CFr1);

//Finding and printing yougest players

List<Player> youngest = TaskUtils.Youngest(allPlayers);

Console.WriteLine("Jauniausi krepšininkai:");

InOutUtils.PrintYoungestPlayers(youngest);

Console.WriteLine();

//Finding and printing players who play in Žalgiris

List<Player> inTheTeam = TaskUtils.Zalgiris(allPlayers, "Žalgiris");

Console.WriteLine("Krepšininkai žaidę Žalgiryje:");

InOutUtils.PrintClubPlayers(inTheTeam);

Console.WriteLine();

//Printing players who celebrates their birthdays of a given time frame

List<Player> whoCelebrates = TaskUtils.CelebratesBirthDays(allPlayers);

InOutUtils.PrintToCsv(CFr2, whoCelebrates);

}

}

}

## Pradiniai duomenys ir rezultatai

Jonas;Valančiūnas;2002-07-28;180;14;Žalgiris;pakviestas;kapitonas

Marius;Grigonis;2002-08-28;179;24;Rytas;pakviestas;žaidėjas

-------------------------------------------------------------------------------------------------------------------------

| Vardas | Pavardė | Gimimo data | Žaidėjo ūgis | Numeris | Klubas | Ar pakviestas | Komandos kapitonas ar ne |

-------------------------------------------------------------------------------------------------------------------------

| Jonas | Valančiūnas | 2002-07-28 | 180 | 14 | Žalgiris | True | True |

| Marius | Grigonis | 2002-08-28 | 179 | 24 | Rytas | True | False |

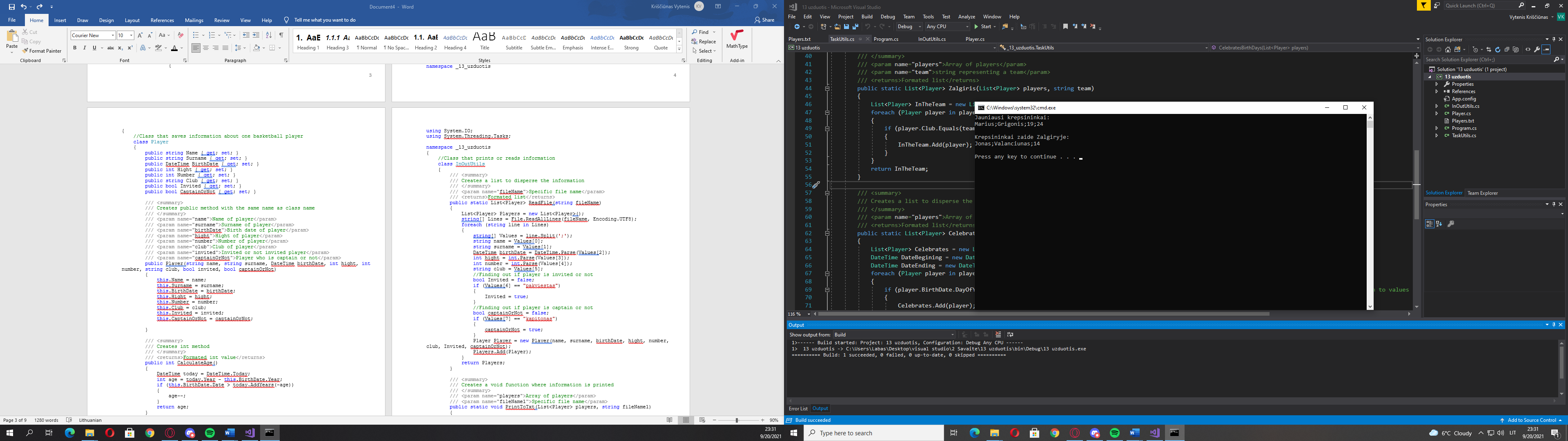
-------------------------------------------------------------------------------------------------------------------------

Jauniausi krepsininkai:

Marius;Grigonis;19;24

Krepsininkai zaide Zalgiryje:

Jonas;Valanciunas;14



**1 Pav.** Atspausdinti rezultatai ekrane

## Dėstytojo pastabos